

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full “License and Conditions of Use” notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

Annex C. Reference Architecture for Institutional Integration of the OncoBalance Model, Python Implementation, and Explanatory Thematic Scenarios

This annex proposes a general reference architecture for institutional integration of the OncoBalance Model and includes a Python implementation skeleton designed to facilitate adoption in hospitals, research centers, and multi-institutional oncology networks. It also adds thematic paragraphs—historical, case-oriented, and simulation-oriented—to illustrate the explanatory power of the model beyond isolated biomarker interpretation. The architecture is designed to be compatible with current interoperability patterns based on FHIR, OMOP, molecular reporting workflows, and explainable AI support under clinical supervision. [file:150][web:228][web:229][web:231][web:238]

1. General reference architecture

A practical institutional architecture for OncoBalance should be organized in six layers:

1. **Clinical and molecular data sources:** EHR, pathology, sequencing pipelines, RNA-seq workflows, methylation assays, liquid-biopsy platforms, and imaging-derived or clinician-entered metadata. FHIR is now widely used in health information technology and provides a strong transport layer for heterogeneous clinical data. [web:228][web:230]
2. **Interoperability and harmonization layer:** ingestion through HL7 FHIR APIs, extraction of oncology data elements, and transformation into research-grade structures. Current HL7 and OHDSI initiatives explicitly support alignment between FHIR transport and OMOP-based research models. [web:229][web:231][web:233]
3. **Canonical storage layer:** OMOP or OMOP-compatible oncology warehouses to support reproducible analytics, versioned cohorts, and research-scale reuse across institutions. FHIR-to-OMOP mappings are increasingly recognized as a robust path to interoperable research implementations. [web:229][web:231][web:239]

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.

No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.

See full “License and Conditions of Use” notice for details.

In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.

Author: Carlos Daniel Borrego Romero

4. **Feature engineering layer:** transformation of raw or semi-structured data into OncoBalance variables, including TMB, oncogenic expression scores, epigenetic-disruption indices, repair-capacity scores, immune-activity indices, and longitudinal change variables. [file:150]
5. **Model and AI orchestration layer:** execution of the OncoBalance score, explainability modules, calibration logic, and optional AI-driven harmonization or anomaly detection. Multi-omics AI reviews emphasize that this layer is where cross-modal structure becomes clinically useful. [web:211][web:238]
6. **Clinical reporting and governance layer:** structured outputs to molecular tumor boards, physician-facing dashboards, audit logs, version control, and intended-use constraints. Multi-omics feasibility studies show that clinically meaningful turnaround and MTB integration are already achievable with current technology. [web:238]

The reference flow is therefore: **FHIR/EHR + molecular pipelines → harmonization → OMOP/research store → feature extraction → OncoBalance engine → explainable clinical report**. This sequence is feasible today and aligns with the current trajectory of interoperable precision oncology. [web:228][web:231][web:236][web:238]

2. Functional design principles

Several design principles are essential for institutional deployment. First, the architecture should be modular so that hospitals can begin with a reduced version of the model—for example, genomics plus expression—before expanding toward methylation and longitudinal liquid-biopsy integration. Second, every derived variable should be versioned and traceable back to source data, because medical models require reproducibility and auditability. Third, AI should be used as an integration accelerator, not as an opaque substitute for clinical judgment. [file:150][web:212][web:229][web:238]

A fourth principle is that interoperability must be semantic as well as syntactic. FHIR enables structured exchange, but real institutional reuse also requires consistent ontology mappings, transformation rules, and cohort definitions. This is one reason FHIR-to-OMOP workflows are increasingly important for translational oncology infrastructures. [web:229][web:231][web:239]

3. Python reference implementation

The following Python architecture is a reference implementation skeleton designed for institutional adaptation. It does not claim to be a final production system; rather, it provides a modular scaffold

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.

No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.

See full “License and Conditions of Use” notice for details.

In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.

Author: Carlos Daniel Borrego Romero

showing how data ingestion, mapping, feature generation, scoring, and reporting can be organized in a clinically realistic pipeline. The structure is intentionally explicit so that each institution can replace local components without breaking the overall logic. [file:150][web:236]

```
from dataclasses import dataclass, field
from typing import Dict, List, Optional, Any
import math

@dataclass
class PatientRecord:
    patient_id: str
    tumor_type: str
    clinical: Dict[str, Any] = field(default_factory=dict)
    genomics: Dict[str, Any] = field(default_factory=dict)
    expression: Dict[str, Any] = field(default_factory=dict)
    methylation: Dict[str, Any] = field(default_factory=dict)
    immune: Dict[str, Any] = field(default_factory=dict)
    liquid_biopsy: List[Dict[str, Any]] = field(default_factory=list)

class DataIngestionLayer:
    def from_fhir_bundle(self, bundle: Dict[str, Any]) -> PatientRecord:
        patient_id = bundle.get("patient_id", "unknown")
        tumor_type = bundle.get("tumor_type", "unspecified")
        return PatientRecord(
            patient_id=patient_id,
            tumor_type=tumor_type,
            clinical=bundle.get("clinical", {}),
            genomics=bundle.get("genomics", {}),
            expression=bundle.get("expression", {}),
            methylation=bundle.get("methylation", {}),
            immune=bundle.get("immune", {}),
            liquid_biopsy=bundle.get("liquid_biopsy", []),
        )

class Normalizer:
    @staticmethod
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.

No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.

See full "License and Conditions of Use" notice for details.

In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.

Author: Carlos Daniel Borrego Romero

```
def bounded_scale(value: float, lower: float, upper: float) -> float:
    if upper <= lower:
        raise ValueError("upper must be greater than lower")
    x = (value - lower) / (upper - lower)
    return max(0.0, min(1.0, x))

@staticmethod
def invert(x: float) -> float:
    return max(0.0, min(1.0, 1.0 - x))

class FeatureEngineeringLayer:
    def __init__(self, config: Dict[str, Any]):
        self.config = config

    def compute_tmb_score(self, genomics: Dict[str, Any]) -> float:
        tmb = float(genomics.get("tmb_mut_per_mb", 0.0))
        cfg = self.config["tmb"]
        return Normalizer.bounded_scale(tmb, cfg["lower"], cfg["upper"])

    def compute_expression_score(self, expression: Dict[str, Any]) -> float:
        signature = float(expression.get("oncogenic_signature_score", 0.0))
        cfg = self.config["expression"]
        return Normalizer.bounded_scale(signature, cfg["lower"], cfg["upper"])

    def compute_epigenetic_score(self, methylation: Dict[str, Any]) -> float:
        epi = float(methylation.get("epigenetic_disruption_score", 0.0))
        cfg = self.config["epigenetic"]
        return Normalizer.bounded_scale(epi, cfg["lower"], cfg["upper"])

    def compute_repair_score(self, genomics: Dict[str, Any], expression: Dict[str, Any]) -> float:
        hrd_raw = float(genomics.get("hrd_score", 0.0))
        repair_pathway = float(expression.get("repair_signature_score", 0.0))
        hrd_cfg = self.config["hrd"]
        expr_cfg = self.config["repair_expression"]
        hrd_deficiency = Normalizer.bounded_scale(hrd_raw, hrd_cfg["lower"], hrd_cfg["upper"])
        repair_expr = Normalizer.bounded_scale(repair_pathway, expr_cfg["lower"], expr_cfg["upper"])
        return max(0.0, min(1.0, 0.5 * Normalizer.invert(hrd_deficiency) + 0.5 * repair_expr))

    def compute_immune_score(self, immune: Dict[str, Any]) -> float:
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.

No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.

See full “License and Conditions of Use” notice for details.

In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.

Author: Carlos Daniel Borrego Romero

```
cd8 = float(immune.get("cd8_score", 0.0))
treg = float(immune.get("treg_score", 0.0))
nk = float(immune.get("nk_score", 0.0))
numerator = cd8 + nk
denominator = numerator + treg + 1e-6
ratio = numerator / denominator
return max(0.0, min(1.0, ratio))

def compute_delta_b(self, liquid_biopsy: List[Dict[str, Any]], score_fn) -> Optional[float]:
    if len(liquid_biopsy) < 2:
        return None
    ordered = sorted(liquid_biopsy, key=lambda x: x["time"])
    t1, t2 = ordered[-2], ordered[-1]
    b1 = score_fn(t1)
    b2 = score_fn(t2)
    dt = float(t2["time"]) - t1["time"]
    if dt <= 0:
        return None
    return (b2 - b1) / dt

class OncoBalanceEngine:
    def __init__(self, weights: Dict[str, float], k: float, theta: float, epsilon: float = 0.1):
        self.weights = weights
        self.k = k
        self.theta = theta
        self.epsilon = epsilon

    def compute_hq(self, tmb: float, egene: float, eepe: float) -> float:
        return (
            self.weights["tmb"] * tmb
            + self.weights["expression"] * egene
            + self.weights["epigenetic"] * eepe
        )

    def compute_b(self, hq: float, repair: float, immune: float) -> float:
        return hq / (repair + immune + self.epsilon)

    def compute_probability_score(self, b: float) -> float:
        return 1.0 / (1.0 + math.exp(-self.k * (b - self.theta)))
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.

No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.

See full "License and Conditions of Use" notice for details.

In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.

Author: Carlos Daniel Borrego Romero

```
class ReportingLayer:
    def build_report(self, patient: PatientRecord, features: Dict[str, float], b: float, p: float, delta_b: Optional[float]) -> Dict[str, Any]:
        return {
            "patient_id": patient.patient_id,
            "tumor_type": patient.tumor_type,
            "oncobalance_features": features,
            "balance_index": round(b, 4),
            "malignancy_risk_score": round(p, 4),
            "delta_b": None if delta_b is None else round(delta_b, 4),
            "interpretation": self._interpret(features, b, p, delta_b),
        }

    def _interpret(self, features, b, p, delta_b):
        dominant = max(features, key=features.get)
        text = f"Dominant destabilizing contributor: {dominant}. Balance index={b:.4f}, malignancy-risk score={p:.4f}."
        if delta_b is not None:
            text += f"Longitudinal imbalance velocity={delta_b:.4f}."
        return text

def run_pipeline(bundle: Dict[str, Any], config: Dict[str, Any], weights: Dict[str, float], k: float, theta: float) -> Dict[str, Any]:
    ingestion = DataIngestionLayer()
    patient = ingestion.from_fhir_bundle(bundle)

    fe = FeatureEngineeringLayer(config)
    tmb_score = fe.compute_tmb_score(patient.genomics)
    egene_score = fe.compute_expression_score(patient.expression)
    eepi_score = fe.compute_epigenetic_score(patient.methylation)
    repair_score = fe.compute_repair_score(patient.genomics, patient.expression)
    immune_score = fe.compute_immune_score(patient.immune)

    engine = OncoBalanceEngine(weights=weights, k=k, theta=theta)
    hq = engine.compute_hq(tmb_score, egene_score, eepi_score)
    b = engine.compute_b(hq, repair_score, immune_score)
    p = engine.compute_probability_score(b)

    report = ReportingLayer().build_report(
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.

No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.

See full “License and Conditions of Use” notice for details.

In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.

Author: Carlos Daniel Borrego Romero

```
patient=patient,
features={
    "tmb": tmb_score,
    "expression": egene_score,
    "epigenetic": eepe_score,
    "repair": repair_score,
    "immune": immune_score,
},
b=b,
p=p,
delta_b=None,
)
return report
```

This code reflects the main institutional logic of the model: source ingestion, normalization, feature definition, balance computation, and structured reporting. In a real institutional deployment, the FHIR ingestion layer could be connected to hospital APIs or exported bundles, the feature-engineering layer could be linked to molecular pipelines, and the reporting layer could feed molecular tumor board dashboards or research registries. [web:228][web:229][web:236][web:238]

4. Historical

Historically, oncology has moved through successive explanatory eras: morphology, histopathology, molecular genetics, targeted biomarkers, and now multi-omics integration. The explanatory power of OncoBalance lies in the fact that it does not reject these prior stages, but rather reorganizes them into a systems framework in which cancer is interpreted as progressive failure of biological equilibrium. In this sense, the model can be read as a historical synthesis: pathology identifies the lesion, genomics identifies the destabilizing burden, immunology reveals the defensive context, and systems modeling explains how these forces interact over time. [file:150][web:148][web:238]

5. Case-oriented thematic paragraph

A case-oriented illustration shows why the model has explanatory value beyond isolated biomarkers. Consider two patients with apparently similar risk by conventional staging: both have localized tumors and comparable clinical burden, yet one has moderate TMB with severe repair collapse and weak immune surveillance, whereas the other has similar TMB but preserved repair and stronger immune

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full “License and Conditions of Use” notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

infiltration. Traditional approaches may classify them similarly at baseline, but OncoBalance explains why their biological trajectories may diverge: the first case sits in a deeper instability regime, while the second remains partially buffered by protection mechanisms. The model therefore adds a mechanistic narrative to what would otherwise be a superficially similar clinical picture. [file:150]

6. Simulation-oriented

A simulation-oriented perspective further highlights the model's power. If one perturbs the system by progressively increasing oncogenic burden while keeping repair and immune terms stable, the balance index rises smoothly and then enters a higher-risk regime through the logistic transition. If, however, one instead simulates gradual immune collapse with only modest increase in oncogenic burden, the model can still predict substantial escalation of malignancy-risk. This matters because it shows that malignant transition is not explained by a single causal route: multiple biological deterioration pathways can converge on the same final clinical state, and the model makes these alternative trajectories analytically visible. [file:150]

7. Concluding integration

Annex C demonstrates that the OncoBalance Model can be translated into a technically realistic institutional architecture using standards and tools that already exist, including FHIR-based ingestion, OMOP-oriented harmonization, Python-based orchestration, and explainable reporting for molecular tumor boards. At the same time, the historical, case-oriented, and simulation-oriented paragraphs show that the model is not only computationally implementable but also conceptually powerful: it offers a richer explanatory language for understanding how genomic burden, failed repair, epigenetic disruption, and immune decline interact in the emergence and progression of cancer. [file:150][web:228][web:229][web:238]

LICENSE AND CONDITIONS OF USE.

The content is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) license:
<https://creativecommons.org/licenses/by-nc-sa/4.0/>
By using this content and model, you agree to:
Provide appropriate credit, include a link to the above license, and indicate if changes were made.
Not use the material or any derivative works for commercial purposes.
Distribute any adaptations or derivative works only under the same CC BY-NC-SA 4.0 license.
Not apply legal terms or technological measures that legally restrict others from doing anything the license permits.
Data logging and trade secrets policy.

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.

No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.

See full "License and Conditions of Use" notice for details.

In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.

Author: Carlos Daniel Borrego Romero

The design, implementation, and deployment of this model, as well as any derivatives, do not authorize the collection, storage, or exploitation of IP addresses, unique device identifiers, or other technical traceability data for commercial profiling, targeted advertising, or any other economic exploitation.

Any system implementing this model must limit the logging and retention of IP addresses and other identifiable data strictly to what is necessary to:

(a) maintain technical security and service integrity; and

(b) comply with applicable legal obligations (for example, any minimum log retention required by law).

It is prohibited to use this model or its derivatives to:

capture, store, or exploit trade secrets, proprietary know-how, or confidential information of users or third parties for any purpose other than providing the technical service described in this documentation; or train or improve closed, commercial systems using data that contains confidential information of third parties without their prior, explicit, written consent.

---+EOD+---